Our Reference: 042309.P12371   *Patent*

METHOD AND APPARATUS FOR SOFTWARE SELECTION OF PROTECTED REGISTER SETTINGS

Inventors: Opher D. Kahn
Alon Naveh

Respectfully submitted,

BLAKELY SOKOLOFF TAYLOR & ZAFMAN LLP
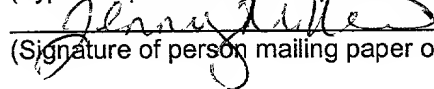
_____
Lawrence E. Lycke
Reg. No. 38,540

Serial/Patent No.: **Not yet assigned**   Filing/Issue Date: ____**Herewith**____
Client: ____**Intel Corporation**____
Title: __**METHOD AND APPARATUS FOR SOFTWARE SELECTION OF PROTECTED REGISTER SETTINGS**__
BSTZ File No.: __**042390.P12371**__   Atty/Secty Initials: ____**LEL/jem**____
Date Mailed: __**November 14, 2001**__   Docket Due Date: ____
The following has been received in the U.S. Patent & Trademark Office on the date stamped hereon:

- ☐ Amendment/Response (_____ pgs.)
- ☐ Appeal Brief (_____ pgs.) (in triplicate)
- ■ Application - Utility (_**23**_ pgs., with cover and abstract)
- ☐ Application - Rule 1.53(b) Continuation (_____ pgs.)
- ☐ Application - Rule 1.53(b) Divisional (_____ pgs.)
- ☐ Application - Rule 1.53(b) CIP (_____ pgs.)
- ☐ Application - Rule 1.53(d) CPA Transmittal (_____ pgs.)
- ☐ Application - Design (_____ pgs.)
- ☐ Application - PCT (_____ pgs.)
- ☐ Application - Provisional (_____ pgs.)
- ■ Assignment and Cover Sheet
- ■ Certificate of Mailing
- ■ Declaration & POA (_**6**_ pgs.)
- ☐ Disclosure Docs & Org & Copy of Inventor's Signed Letter(_____pgs)
- ■ Drawings: _**3**_ # of sheets includes _**4**_ figures

- ■ Express Mail No.**EL861982967US**   ■ Check No.**1798**
- ☐ _____ Month(s) Extension of Time   Amt: **$1,212.00**
- ☐ Information Disclosure Statement & PTO-1449 (__ pgs.)   ☐ Check No._____
- ☐ Issue Fee Transmittal   Amt: _____
- ☐ Notice of Appeal
- ☐ Petition for Extension of Time
- ☐ Petition for _____
- ■ Postcard
- ☐ Power of Attorney (_____ pgs.)
- ☐ Preliminary Amendment (_____ pgs.)
- ☐ Reply Brief (_____ pgs.)
- ☐ Response to Notice of Missing Parts
- ☐ Small Entity Declaration for Indep. Inventor/Small Business
- ■ Transmittal Letter, in duplicate   **2 pages each**
- ■ Fee Transmittal, in duplicate   **2 pages each**

- ■ Other: ____**Certificate of mailing with copy of return postcard signed by attorney.**____

APPLICATION FOR UNITED STATES LETTERS PATENT

For

METHOD AND APPARATUS FOR SOFTWARE SELECTION OF PROTECTED REGISTER SETTINGS

Inventor:

Opher D. Kahn
Alon Naveh

Prepared by:

BLAKELY SOKOLOFF TAYLOR & ZAFMAN LLP
12400 Wilshire Boulevard
Los Angeles, CA 90025-1026
(206) 292-8600

Attorney's Docket No.: 042390.P12371

1

METHOD AND APPARATUS FOR SOFTWARE SELECTION OF PROTECTED REGISTER SETTINGS

FIELD OF THE INVENTION

[0001] The field of invention relates to electrical circuitry in general; and, more specifically, to control register circuits.

BACKGROUND

[0002] Memory controller circuits can be used in a variety of computer systems (*e.g.*, desktop personal computers, notebook computers, personal digital assistants, *etc.*) to facilitate the computer system's processor in accessing memory chips. For example, a memory controller can have an interface for connecting to one or more synchronous dynamic RAM (SDRAM) chips. The memory controller uses this memory interface to route data between the processor and RAM chips and to send address and control signals to the RAM chips.

[0003] A memory controller typically includes a set of control registers to store data needed for operations performed by the memory controller. This data is stored in the control registers so that the data can be changed by the basic input output system (BIOS) or software such as, for example, the operating system (OS) or other driver software.

[0004] The data stored in some control registers is hardware protected after being loaded by the BIOS during a hard or full reset operation. For example, some registers store threshold values or settings that are used in controlling a voltage or

the temperature of the memory controller. The memory controller protects such registers so that a user or a virus cannot change the values of these thresholds or settings to a level that can cause faulty operation or even damage the chip.

[0005]   However, this register protection system lacks flexibility in that the values loaded into these protected registers are typically set for a worst-case scenario. For example, the settings may include the duration of a throttling operation when the memory controller's temperature gets too high. This duration would be set for the worst-case heat removal rate. Unfortunately, the memory controller's performance is degraded for the duration of the throttling operation.

[0006]   In some applications, the heat removal rate can change depending on the application's operational mode. For example, the memory controller may be used in a notebook computer, which can be operated in an undocked mode (*i.e.*, using battery power) or in a docked mode (*i.e.*, docked in a docking station that has its own power source). In the undocked mode, the notebook computer may be configured to turn off a fan to conserve power, resulting a relatively low heat removal rate. Thus, when the fan is off, the throttling duration should be relatively lengthy. In contrast, when the fan is on, the heat removal rate is relatively high and the throttling duration can be shorter. However, because the OS or application software cannot change the data stored in the protected registers by the BIOS after a full reset, the throttling duration is loaded for the worst case (*i.e.*, the duration needed to cool the chip when the fan is off).

BRIEF DESCRIPTION OF THE DRAWINGS

[0007] Non-limiting and non-exhaustive embodiments of the present invention are described with reference to the following figures, wherein like reference numerals refer to like parts throughout the various views unless otherwise specified.

[0008] Figure 1 is a simplified block diagram illustrating a memory controller as part of a computer system, according to one embodiment of the present invention.

[0009] Figure 2 is a flow diagram illustrating the operation of the memory controller of Figure 1, according to one embodiment of the present invention.

[0010] Figure 3 is a simplified block diagram illustrating a software-controlled protected register unit, according to one embodiment of the present invention.

[0011] Figure 4 is a block diagram illustrating an implementation of the software-controlled protected register unit of Figure 3, according to one embodiment of the present invention.

DETAILED DESCRIPTION

[0012]   Figure 1 illustrates in simplified form a computer system 10 with a memory controller 11, according to one embodiment of the present invention. In addition, computer system 10 includes a processor 13 and a memory 15. In accordance with the present invention, memory controller 11 includes a software-controlled protected register unit 17 (also referred to herein as register unit 17). Register unit 17 is described further below.

[0013]   The elements of computer system 10 are interconnected as follows. Processor 13 is connected to memory controller 11 through a bus 18. Memory 15 is connected to memory controller 11 through a bus 19. In this embodiment, processor 13 can load control registers of register unit 17 via bus 18.

[0014]   Figure 2 illustrates operations performed by register unit 17. Referring to Figures 1 and 2, register unit 17 operates as follows. After a full reset operation, the control registers of memory controller 11 are loaded by the BIOS. In this embodiment, the control registers of memory controller 11 reside in register unit 17. The control registers are loaded by the BIOS during execution by processor 13 of computer system 10. In other embodiments, some of these control registers may be implemented with non-volatile memory, which need not be loaded after the full reset operation. The control registers include protected and non-protected control registers. This operation is represented by a block 21 of Figure 2.

[0015]   Some of the control registers of memory controller 11 are protected registers. In this embodiment, the protected register have hardware mechanisms

that prevent changing of the data stored in the protected registers once the protected registers are locked. The protected registered are unlocked only by a full reset, and are typically locked as part of the BIOS initialization sequence. As previously described, protected registers are typically used to store "sensitive" data such as threshold levels or settings used by other units (*e.g.*, see Figure 3). The term "sensitive" is used in this context to refer to data that, if outside predetermined acceptable ranges, can degrade the operation of or cause damage to memory controller 11 when used by these other units of memory controller 11. For example, this data can be temperature trip points and/or clock throttling settings for a temperature control unit, or voltage settings for different power saving modes provided by a power conservation unit, *etc.*

[0016]   In one embodiment, the protected registers are locked by the BIOS after the full reset operation. For example, in one embodiment, the BIOS may lock the protected registers by setting "lock" bits of the protected registers. In other embodiments, the locking operation will depend on the design of the protected registers. This operation is represented by a block 23 in Figure 2.

[0017]   In a further refinement, the operation of block 23 (*i.e.*, locking the protected registers) may be performed simultaneously with block 21 (*i.e.*, loading the control registers). For example, in one embodiment, the BIOS may perform a single operation to load a protected register's data along with setting the lock bit.

[0018]   In accordance with the present invention, the protected registers may include sets of two or more protected registers that can each provide data to a

corresponding unit of the aforementioned other units of memory controller 11. Each set of these protected registers is loaded with data that are optimized for use during predetermined circumstances. The data of each of set of related protected registers is predetermined to be "safe" for use by its corresponding unit (*i.e.*, the unit will not damage memory controller 11 when using the data stored in its corresponding the set of protected registers). This operation is represented by a block 25 in Figure 2.

[0019]   For example, a set of related protected registers may store durations of throttling operations to be used by a thermal control unit (*e.g.*, see Figure 3) of memory control unit 11. One protected register of this set may store a duration for use by the thermal control unit when a cooling fan of computer system 10 is operating; whereas another protected register of the set may store a duration for use when the cooling fan is not operating. In one embodiment, software such as the OS or driver software can control register unit 17 to select the appropriate protected register (locked during block 23) to use under the current circumstances. Continuing the above throttling example, the OS or application software can detect whether the aforementioned cooling fan is operating and, depending on the result, select the appropriate locked register to be used by the thermal control unit.

[0020]   The data stored in the selected locked register is then outputted to the associated unit of memory controller 11. In some embodiments, the data stored in the selected locked register is accessed by its associated unit when needed. Continuing the throttling example above, the thermal control unit may access the selected locked register to load the stored value into a counter that determines the duration of the throttling operation. For example, the thermal unit may access this

7

locked register when the temperature of memory controller 11 reaches a threshold value (which can also be stored in a protected register). This operation is represented by a block 27 in Figure 2.

[0021]　The selection of which locked register is being used can then be changed by software such as the OS or driver software. If the software is to select a different locked register, the operational flow returns to block 25. However, if the software is maintain the selection of the current selected locked register, the operational flow returns to block 27.

[0022]　This embodiment of register unit 17 allows software to select between two of more protected control registers to provide sensitive data to an unit of memory controller 11. This selection feature advantageously provides flexibility in the use of sensitive control registers while preventing the sensitive control registers from being loaded with unsafe values by the software. Further, although a memory controller application is described above, other embodiments of register unit 17 may be used in other types of circuits that may have protected registers to store sensitive data such as, for example, processors, microcontrollers, input/output (I/O) controllers, etc.

[0023]　Figure 3 illustrates an implementation of register unit 17 (Figure 1), according to one embodiment of the present invention. In this embodiment, register unit 17 includes protected control registers 31 and a selector 33. Protected control registers 31 include one or more sets of protected control registers that are associated with units 34 of memory controller 11. In this exemplary embodiment, units 34 include a thermal control unit 35 and a power conservation unit 36. For

example, protected control registers 31 may include a first set of protected control registers for storing throttling durations, a second set for storing temperature thresholds, and a third set for storing memory transfer bandwidth thresholds, all of which are associated with thermal control unit 35. In addition, protected control registers 31 may include a fourth set of control registers for storing voltage thresholds that are used by power conservation unit 36. Although Figure 3 shows units 34 having only two units in this embodiment, units 34 may include additional units that use protected registers.

[0024] The elements of this embodiment of register unit 17 are interconnected as follows. Protected control registers 31 are connected to input ports of selector 33 via a line 37. Protected control registers 31 may also include control registers that are not connected to selector 33. Although only a single line is shown in Figure 3, line 37 includes, for each protected control register connected to selector 33, a set of conductive interconnect for providing the output of that protected control register to a corresponding port of selector 33. In addition, in this embodiment, selector 33 is connected to thermal control unit 35 and power conservation unit 36 via lines 38 and 39, respectively. Line 38 includes conductive interconnect for connecting, in effect, the output of one control register of protected control registers 31 to thermal control unit 35. In particular, this one control register would be one of a set of control registers associated with thermal control unit 35. Similarly, line 39 includes conductive interconnect for connecting, in effect, the output of one control register of protected control registers 31 to power conservation unit 36. This one control

register would be one of a set of control registers associated with power conservation unit 36.

[0025] In operation, selector 33 is configured to select a protected register of each set of control registers of protected control registers 31 that are associated with units 34. For example, a set of control registers may store throttling durations for use by thermal control unit 35 during various circumstances. Selector 33 selects the throttling duration stored by an appropriate one of the protected control register of this set to provide to thermal control unit 35. Thus, if a cooling fan is operating to cool the chip, selector 33 may be configured to select the protected control register storing a relatively short throttling period. Further, as previously described, software can reconfigure selector 33 to select a different protected control register (of the set) in response to different conditions or user input.

[0026] Figure 4 illustrates selector 33 (Figure 3), according to one embodiment of the present invention. In this embodiment, selector 33 includes a multiplexer (or other switch unit) and a non-protected register for each aforementioned set of protected control registers associated with units 34 (Figure 3). In particular, selector 33 includes a multiplexer 41-1 and non-protected register 42-1 for one set of protected control registers that are associated with one unit of units 34; a multiplexer 41-2 and a non-protected register 42-2 for another set of protected control registers that are associated with another unit of units 34; and so on for each set of protected control registers associated with units 34. In this embodiment, non-protected registers 42-1, 42-2 and so on are control registers that are part of register unit 17 (Figure 3). For example, non-protected registers 42-1, 42-2 and so on can

be essentially identical to protected control registers 31 except for having lock bits that are not set by the BIOS. In other embodiments, these non-protected registers need not have a lock bit.

[0027] The elements of this embodiment of selector 33 are interconnected as follows. Multiplexer 41-1 has input ports connected to output ports of protected control registers $43\text{-}1_1$ to $43\text{-}1_X$, via lines $44\text{-}1_1$ through $44\text{-}1_X$, respectively. In this exemplary embodiment, lines $44\text{-}1_1$ through $44\text{-}1_X$ are each N bits wide. Multiplexer 41-1 also has a control port connected to the output port of non-protected register 42-1 via a line 45-1. In one embodiment, line 45-1 is R bits wide, with 2^R being greater than or equal to X so the data stored in non-protected register 42-1 can be coded to select one of protected control registers $43\text{-}1_1$ through $43\text{-}1_X$. Multiplexer 41-1 has an output port connected to line 38, which in this embodiment is also N bits wide, matching the output ports of protected control registers $43\text{-}1_1$ through $43\text{-}1_X$.

[0028] Similarly, multiplexer 41-2 has input ports connected to output ports of protected control registers $43\text{-}2_1$ to $43\text{-}2_Y$, via lines $44\text{-}2_1$ through $44\text{-}2_Y$, respectively. Lines $44\text{-}2_1$ through $44\text{-}2_Y$ are each M bits wide. Multiplexer 41-2 also has a control port connected to the output port of non-protected register 42-2 via a line 45-2. In this embodiment, line 45-2 is Q bits wide, with 2^Q being greater than or equal to Y so the data stored in non-protected register 42-2 can select one of protected control registers $43\text{-}2_1$ through $43\text{-}2_Y$. Multiplexer 41-2 has an output port connected to line 39, which in this embodiment is also M bits wide, matching the output ports of protected control registers $43\text{-}2_1$ through $43\text{-}2_Y$. Other sets of protected control

registers have corresponding multiplexers and non-protected registers that are similarly interconnected.

[0029] In operation, the BIOS loads protected control registers 31 with predetermined "safe" data right after a full reset operation. As previously described, protected control registers 31 include sets of protected control registers storing data for use by associated units of units 34. In addition, in this embodiment, the BIOS sets a lock bit of each of protected control registers 31 so that software cannot change the safe data.

[0030] In addition, the BIOS loads non-protected registers 42-1, 42-2 and so on with data to select safe data (stored in protected control registers 31) to be provided to units 34. In particular, each of these non-protected registers is loaded with data that is received by its corresponding multiplexer (*i.e.*, one of multiplexers 41-1, 41-2 and so on). Responsive to this data, each multiplexer couples one of its corresponding set of protected control registers to its corresponding unit of units 34.

[0031] Further, for each of these non-protected registers, software can load other data in the non-protected register in response, for example, to a change in conditions, user input, operational mode, *etc.* The software may be configured to select another of the protected control register within the same set of protected control registers to provide data to the corresponding unit that is appropriate for the new conditions, user input, operational mode, *etc.* Thus, continuing the above example, software changes the data stored in non-protected register 42-1 to change

12

the throttling duration in response to a change in chip cooling rate caused by a change cooling fan operation.

[0032]   Embodiments of a software controlled protected register unit are described herein. In the above description, numerous specific details are set forth to provide a thorough understanding of embodiments of the invention. One skilled in the relevant art will recognize, however, that the invention can be practiced without one or more of the specific details, or with other methods, components, materials, *etc.* In other instances, well-known structures, materials, or operations are not shown or described in detail to avoid obscuring aspects of the invention.

[0033]   Reference throughout this specification to "one embodiment" or "an embodiment" means that a particular feature, structure, or characteristic described in connection with the embodiment is included in at least one embodiment of the present invention. Thus, the appearances of the phrases "in one embodiment" or "in an embodiment" in various places throughout this specification are not necessarily all referring to the same embodiment.   Furthermore, the particular features, structures, or characteristics may be combined in any suitable manner in one or more embodiments.

[0034]   In addition, embodiments of the present description may be implemented not only within a semiconductor chip but also within machine-readable media. For example, the designs described above may be stored upon and/or embedded within machine readable media associated with a design tool used for designing semiconductor devices.   Examples include a netlist formatted in the VHSIC

13

Hardware Description Language (VHDL) language, Verilog language or SPICE language. Some netlist examples include: a behavioral level netlist, a register transfer level (RTL) netlist, a gate level netlist and a transistor level netlist. Machine-readable media also include media having layout information such as a GDS-II file. Furthermore, netlist files or other machine-readable media for semiconductor chip design may be used in a simulation environment to perform the methods of the teachings described above.

[0035]   Thus, embodiments of this invention may be used as or to support a software program executed upon some form of processing core (such as the CPU of a computer) or otherwise implemented or realized upon or within a machine-readable medium. A machine-readable medium includes any mechanism for storing or transmitting information in a form readable by a machine (*e.g.*, a computer). For example, a machine-readable medium can include such as a read only memory (ROM); a random access memory (RAM); a magnetic disk storage media; an optical storage media; and a flash memory device, *etc.* In addition, a machine-readable medium can include propagated signals such as electrical, optical, acoustical or other form of propagated signals (*e.g.*, carrier waves, infrared signals, digital signals, *etc.*).

[0036]   In the foregoing specification, the invention has been described with reference to specific exemplary embodiments thereof. It will, however, be evident that various modifications and changes may be made thereto without departing from the broader spirit and scope of the invention as set forth in the appended claims.

The specification and drawings are, accordingly, to be regarded in an illustrative

rather than a restrictive sense.